IN THE SPECIFICATION:

Please insert before Page 1, line 3:

This application claims priority to International Application PCT/US99/02780 ("A System, Method and Article of Manufacture for a Simulation Enabled Accounting Tutorial System"), filed February 8, 1999 and to U.S. Application Ser. No. 09/219,480 ("System, Method and Article of Manufacture for a Simulation Enabled Accounting Tutorial System"), filed December 22, 1998 and granted as U.S. Patent No. 6,029,159.

Page 2, line 21:

~~Figure 10 illustrates~~ Figures 10 and 11 illustrate a journal entry simulation in accordance with a preferred embodiment;

Page 2, line 22:

Figure ~~11~~ 12 illustrates a simulated Bell Phone Bill journal entry in accordance with a preferred embodiment;

Page 2, line 23:

Figure ~~12~~ 13 illustrates a feedback display in accordance with a preferred embodiment;

Page 2, line 24:

Figure ~~13~~ 14 illustrates the steps of the first scenario in accordance with a preferred embodiment;

Page 2, line 25:

Figure ~~14 and~~ 15 ~~illustrate~~ illustrates the steps associated with a build scenario in accordance with a preferred embodiment;

Page 16, second paragraph:

Figure 8 is a Goal-Based Scenario (GBS) ~~GBS~~ display in accordance with a preferred embodiment. The upper right area of the screen shows the account list. There are four types of accounts: Assets, Liabilities & Equity, Revenues, and Expenses. The user clicks on one of the tabs to show the accounts of the corresponding type. The student journalizes a transaction by dragging an account from the account list onto the journal entry Debits or Credits. The student then enters the dollar amounts to debit or credit each account in the entry. In the interface, as in real life, the student can have multi-legged journal entries (i.e., debiting or crediting multiple accounts). A Toolbar 1200 and the first transaction of this Task 1210 appear prominently on the display. The student can move forward and back through the stack of transactions. For each transaction, the student must identify which accounts to debit and which to credit. When the student is done, he clicks the Team button. Figure 9 is a feedback display in accordance with a preferred embodiment. The student may attempt to outsmart the system by submitting without doing anything. The ICAT system identifies that the student has not done a substantial amount of work and returns the administrative feedback depicted in Figure 9. The feedback points out that nothing has been done, but it also states that if the student does some work, the tutor will focus on the first few journal entries. ~~Figure 10 illustrates~~ Figures 10 and 11 illustrate a journal entry simulation in accordance with a preferred embodiment. Figure ~~11~~ 12 illustrates a simulated Bell Phone Bill journal entry in accordance with a preferred embodiment. The journal entry is accomplished by debiting Utilities Expenses and Crediting Cash for $700 each. Figure ~~12~~ 13 illustrates a feedback display in accordance with a preferred embodiment. After attempting to journalize the first three transactions, the student submits his work and receives the feedback depicted in Figure ~~12~~ 13. The feedback starts by focusing the student on the area of work being evaluated. The ICAT states that it is only looking at the first three journal entries. The feedback states that the first two entries are completely wrong, but the third is close. If the student had made large mistakes on each of the first three transactions, then the ICAT may have given redirect feedback, thinking a global error occurred. The third bullet point also highlights how specific the feedback can become, identifying near misses.

Page 16, third paragraph:

Design Scenario-This Scenario illustrates how the tools are used to support conceptual and detailed design of a BusSim application. Figure ~~13~~ 14 illustrates the steps of the first scenario in accordance with a preferred embodiment. The designer has gathered requirements and determined that to support the client's learning objectives, a task is required that teaches journalization skills. The designer begins the design first by learning about journalization herself, and then by using the Knowledge Workbench to sketch a hierarchy of the concepts she want the student to learn. At the most general level, she creates a root concept of 'Journalization'. She refines this by defining sub-concepts of 'Cash related transactions', 'Expense related Transactions', and 'Expense on account transactions'. These are each further refined to whatever level of depth is required to support the quality of the learning and the fidelity of the simulation. The designer then designs the journalization interface. Since a great way to learn is by doing, she decides that the student should be asked to Journalize a set of transactions. She comes up with a set of twenty-two documents that typify those a finance professional might see on the job. They include the gamut of Asset, Expense, Liability and Equity, and Revenue transactions. Also included are some documents that are not supposed to be entered in the journal. These 'Distracters' are sometimes because documents errant documents occur in real life. The designer then uses the Domain Model features in the Knowledge Workbench to paint a Journal. An entity is created in the Domain Model to represent each transaction and each source document. Based on the twenty-two documents that the designer chose, she can anticipate errors that the student might make. For these errors, she creates topics of feedback and populates them with text. She also creates topics of feedback to tell the student when they have succeeded. Feedback Topics are created to handle a variety of situations that the student may cause.

Page 17, third paragraph:

~~Figures 14 and 15 illustrate~~ Figure 15 illustrates the steps associated with a build scenario in accordance with a preferred embodiment. The instructional designer completes the initial interaction and interface designs as seen in the previous Scenario. After low-fi user testing, the Build Phase begins. Graphic artists use the designs to create the bitmaps that will make up the

interface. These include bitmaps for the buttons, tabs, and transactions, as well as all the other screen widgets. The developer builds the interface using the bitmaps and adds the functionality that notifies the Domain Model of student actions. Standard event-driven programming techniques are used to create code that will react to events in the interface during application execution and pass the appropriate information to the Domain Model. The developer does not need to have any deep knowledge about the content because she does not have to build any logic to support analysis of the student actions or feedback. The developer also codes the logic to rebuild the interface based on changes to the domain model. A few passes through these steps will typically be required to get the application communicating correctly with the components. The debug utilities and Regression Test Workbench streamline the process. After the application interface and component communication are functioning as designed, the task is migrated to Usability testing.